1.Terminology Used

For a given item with id = i

$baseSales_i = baseprice_i * baseqty_i$

$Price1sales_i = price1_i * price1qty_i$

and so on……

total_base_price maintains a running sum of minimum expected revenue
total_base_qty maintains a running sum of minimum expected quantity
total_sale_price maintains the running revenue earned by selecting prices
total_sale_qty maintains the running aggregate quantity by selecting prices

2.Algorithm

For each item starting iteration from the one with id = 1
We calculate the *total_base_price* and *total_base_qty* upto that item id.
If selecting $baseSales_i$ for an item does not violate the constraints below, we select the base price. Base price is given first priority because it incurs zero hit.

total_sales_price>=total_base_price ……. (1)

and

total_sales_qty>=total_base_qty, ……. (2)

If selecting base price for an item violates the constraints (1) and (2), we sort the other prices in ascending order of their hits because minimizing hits deserves greater precedence. Starting from the price yielding the smallest hit, we check if adding it violates the constraints (1) and (2). If they do not, we select that price and move onto the next item id because it is enough to satisfy (1) and (2), else check the next price.
If all the prices violate the constraints (1) and (2), we select the price corresponding to max($Price1sales_i$, $Price2sales_i$, $Price3sales_i$, $Price4sales_i$).